A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints

Claudio Contardo

Département de management et technologie, ESG UQÀM
315 Ste-Catherine Est, Montréal (QC), Canada H2X 3X2
e-mail address: contardo.claudio@uqam.ca

This article presents an exact algorithm for the multi-depot vehicle routing problem (MDVRP) under capacity and route length constraints. The MDVRP is formulated using a vehicle-flow and a set-partitioning formulation, both of which are exploited at different stages of the algorithm. The lower bound computed with the vehicle-flow formulation is used to eliminate non-promising edges, thus reducing the complexity of the pricing subproblem used to solve the set-partitioning formulation. Several classes of valid inequalities are added to strengthen both formulations, including a new family of valid inequalities used to forbid cycles of an arbitrary length. To validate our approach, we also consider the capacitated vehicle routing problem (CVRP) as a particular case of the MDVRP, and conduct extensive computational experiments on several instances from the literature to show its effectiveness. The computational results show that the proposed algorithm is competitive against state-of-the-art methods for these two classes of vehicle routing problems, and is able to solve to optimality some previously open instances. Moreover, for the instances that cannot be solved by the proposed algorithm, the final lower bounds prove stronger than those obtained by earlier methods.

1 Introduction

The multi-depot vehicle routing problem (MDVRP) is an important class of vehicle routing problem arising in freight distribution and can be defined as follows. We are given a set of depot locations $\mathcal{D}$ and a set of customer locations $\mathcal{C}$, which are assumed to be disjoint (even if two points share the same physical coordinates, they are still handled as different entities). With every customer $j \in \mathcal{C}$ is associated a demand d_j. With every depot location $i \in \mathcal{D}$ is associated a fleet of size m_i. The fleet is assumed to be homogeneous with all vehicles having the same capacity Q and having to respect a maximum route length of T units. We consider a graph $G = (V, E)$ with $V = \mathcal{D} \cup \mathcal{C}$ and $E = \{\{i, j\} : i, j \in V,\ i \text{ and } j \text{ not both in } \mathcal{D}\}$. With every edge $e \in E$ is associated a travel time t_e. The objective is to select a subset of vehicles and to construct routes that respect the capacity and route length constraints, so as to visit each customer exactly once, at minimum traveling cost. The capacitated vehicle routing problem (CVRP) is a particular case of the MDVRP in which $|\mathcal{D}| = 1$, the number of vehicles to be used is exactly m and the maximum route length constraint is relaxed.

Both problems are $\mathcal{NP}$-hard since they are generalizations of the traveling salesman problem, therefore polynomial-time algorithms are unlikely to exist unless $\mathcal{P} = \mathcal{NP}$ [20]. Despite of the computational complexity of these problems, state-of-the-art heuristic methods can

find near-optimal solutions in a matter of seconds [14, 31].

The literature on exact approaches for the MDVRP is sparse. In fact, most authors have focused on the development of heuristic methods to find good quality solutions quickly [29, 14, 31]. The most recent exact method reporting results on the MDVRP is that of Baldacci and Mingozzi [2]. The method is based on the additive bounding procedure of Christofides et al. [11] applied to several different relaxations of the problem. Ultimately, the set-partitioning formulation of the MDVRP is solved by means of column generation strengthened with the so-called strong capacity constraints and clique inequalities. Note that their modeling approach does not consider the route length constraint and so experiments are conducted only on those instances without such constraint. A problem closely related to the MDVRP is the periodic VRP (PVRP), in which the complete planning horizon is subdivided in periods, and vehicle routes cannot be longer than the length of one period. The MDVRP can be formulated as a PVRP by realizing that different depots can be modeled as multiple periods in the context of a PVRP. Therefore, any algorithm that solves the PVRP can also solve the MDVRP. Baldacci et al. [5] proposed an exact algorithm for the PVRP that generalizes their former method for the MDVRP but, as remarked by the authors, does not improve their earlier results.

With respect to the CVRP, the literature on exact methods is broader. The most efficient exact algorithms for the CVRP are those of Lysgaard et al. [28], Fukasawa et al. [19], Baldacci et al. [3] and Baldacci et al. [6]. Lysgaard et al. [28] developed the most efficient branch-and-cut algorithm for the CVRP. They consider a compact two-index vehicle-flow formulation of the problem and use several classes of valid inequalities for which they provide novel and efficient separation algorithms. Their method is able to consistently solve problems with up to 50 customers. Fukasawa et al. [19] developed the first branch-and-cut-and-price algorithm for the CVRP. They consider a set partitioning formulation in which variables represent vehicle routes satisfying the capacity constraint. The routes are allowed to contain cycles, and the pricing subproblem can be solved efficiently by using dynamic programming. This column generation approach is embedded into a branch-and-cut framework, using several of the valid inequalities introduced in [28]. Their computational results show that their method can scale and solve instances with twice as many customers as the previous exact method of Lysgaard et al. [28]. Baldacci and Mingozzi [2] and Baldacci et al. [6] introduced a new approach to solve the CVRP based on the additive bounding technique introduced by Christofides et al. [11] on which several different relaxations of the problem are solved sequentially in an additive manner. At last, the set-partitioning formulation of the problem is strengthened with strong capacity cuts and clique inequalities, and solved by column generation. The last part of the algorithms reduces to solve a pure integer linear problem with a reduced number of variables. The methods of Baldacci and Mingozzi [2] and Baldacci et al. [6] differ mainly in the SPPRC relaxation used. While the former relies on elementary routes, the latter relies now in the so-called *ng*-routes relaxation, a new pricing subproblem that produces near-elementary routes in a fraction of the computational effort. This relaxation is able to produce near-elementary routes in a fraction of the computational effort used to solve the traditional ESPPRC. While the method of Baldacci and Mingozzi [2] drastically reduced the computing times with respect to the former method of Fukasawa et al. [19], now the method based on the *ng*-routes relaxation proves even faster and is able to solve some instances that the previous method did not.

With respect to algorithmic enhancements for column generation methods for vehicle routing problems, they can be categorized into two main branches. On the one hand, the development of new pricing subproblems aimed to achieve the so-called elementary bound (the lower bound obtained when the set of feasible routes is restricted to those not containing cycles) without solving the elementary shortest path problem with resource constraints (ESPPRC) at every iteration, which is known to be strongly $\mathcal{NP}$-hard. On the other hand, the development of new families of valid inequalities aimed to strengthen the linear relaxations of the set-partitioning formulations of these problems.

In the first category, authors have focused in the development of relaxations of the original ESPPRC to consider routes with cycles. The shortest path problem under resource constraints and without cycles of length one or two (2-*cyc*-SPPRC), introduced by Houck et al. [22] and later used by Desrochers et al. [17] in the context of the vehicle routing problem with time windows (VRPTW) is an example of such relaxation. In the 2-*cyc*-SPPRC, routes are allowed to contain cycles as long as these cycles do not visit the same node twice with a separation of one intermediate node. The shortest path with resource constraints and without cycles of length k for $k \geq 3$ introduced by Irnich and Villeneuve [23] (k-*cyc*-SPPRC) is another example, in which routes are now allowed to contain cycles as long as no route visits a customer twice with less than k customers of separation. The decremental state-space relaxation (DSSR) introduced by Righini and Salani [30] achieves the elementary bound by first relaxing the elementarity constraint, and iteratively imposing elementarity on the nodes with cycles, until no more cycles are detected. The *ng*-routes relaxation (*ng*-SPPRC) introduced by Baldacci et al. [6] is a very efficient relaxation that achieves near-elementary routes in very short computing times. The intuition behind the *ng*-routes relaxation is as follows. During a labeling algorithm, cycles are allowed as long as they do not visit a set of forbidden nodes, which is constructed from predefined sets of neighbors of every node, and that intuitively forces cycles to contain nodes that are far from each other. The set of forbidden nodes is updated after every extension of the labels and is made in such a way that the efficiency of the dynamic programming algorithm is not compromised. As a result, very strong lower bounds can be obtained that in many cases coincide with the elementary bounds. More recently, Contardo et al. [13] introduced the so-called strong degree constraints (SDC), a new family of valid inequalities that are proved to impose partial elementarity. The addition of a strong degree constraint associated to a certain customer imposes that no variable associated to a route having a cycle on that node will be basic in the LP relaxation. The dual variable associated to such constraint can be used to derive a sharper rule than that of classic elementarity.

With respect to the development of new valid inequalities for the CVRP and the MDVRP, Fukasawa et al. [19] were the first to adapt the inequalities used by Lysgaard et al. [28] for the CVRP to the set-partitioning formulation. They showed that these inequalities can be included in the problem without breaking the structure of the pricing subproblem. They used capacity constraints (CC), strengthened comb inequalities (SCI) and framed capacity inequalities (FCI) in their algorithm. Baldacci et al. [3] introduced the so-called strong capacity constraints (SCC), by realizing that the original capacity constraints could be lifted in the particular case of the set-partitioning formulation, yielding much stronger bounds. The addition of a SCC, however, imposes the addition of an extra resource in the labeling algorithm and therefore makes it harder. Jepsen et al. [24] introduced the subset-row inequal-

ities (SRI), a particular family of rank-1 Chvátal-Gomory cuts [21]. They showed that these inequalities can be efficiently separated and included in the pricing subproblems without compromising the overall performance of the algorithm. Contardo et al. [13] introduced the y-strong capacity constraints (y-SCC) and the strong framed capacity inequalities (SFCI) in the context of the capacitated location-routing problem (CLRP). These two inequalities are shown to dominate both the SCC and the FCI, respectively.

In this paper, we propose a new exact method for the MDVRP based on the solution of ad-hoc vehicle-flow and set-partitioning formulations strengthened with several classes of valid inequalities. The set-partitioning formulation is solved by column-and-cut generation, for which the column generation subproblem is solved using the 2-*cyc*-SPPRC, and we use the SDC to impose partial elementarity. Because the SDC may be hard to handle when their associated dual variables are large, we complement their use with the inclusion of a new family of valid inequalities to forbid cycles of an arbitrary length. As a result, our method is able to produce tight lower bounds and to solve to optimality some hard instances of the MDVRP and the CVRP in short computing times, including some previously unsolved instances. Moreover, for the instances that remain unsolved, our algorithm provides tighter lower bounds than existing methods.

The remaining of the article is organized as follows. In Section 2 we introduce the two formulations for the MDVRP used in this paper, namely a compact two-index vehicle-flow formulation and a set-partitioning formulation. In Section 3 we present the valid inequalities used in our method. In Section 4 we present the exact method that solves the MDVRP to optimality. In Section 5 we report our computational results on selected instances from the literature for the CVRP and the MDVRP. We conclude the article in Section 6.

2 Mathematical formulations

In this section, we present the two formulations used in this article to model and solve the MDVRP, namely a vehicle-flow and a set-partitioning formulation. These two formulations of the problem are exploited at different stages of the algorithm. More precisely, the vehicle-flow formulation is used to derive a lower bound quickly and perform variable fixing. The set-partitioning formulation is then considered using the reduced network produced by the variable fixing procedure performed before.

2.1 Vehicle-flow formulation of the MDVRP

For every depot $i \in \mathcal{D}$ and customer $j \in \mathcal{C}$, let y_{ij} be a binary variable equal to 1 iff j is served by a single-vehicle route departing from depot i. For every $e \in E$, we let x_e be a binary variable equal to 1 iff edge e is used by a vehicle route visiting at least two customers. For a customer set $S \subseteq \mathcal{C}$, we let $r(S) = \lceil d(S)/Q \rceil$ be a lower bound on the number of vehicles needed to serve the customers in S due to the capacity constraint, and we let $\rho(S)$ be a lower bound on the number of vehicles needed to serve the customers in S due to the route length constraint. Note that while $r(S)$ can be computed in constant time, $\rho(S)$ can be difficult to compute as it involves the solution of a m-TSP with route length constraints, which is strongly $\mathcal{NP}$-hard. For a node subset $U \subset V$ we let $\delta(U)$ be the cutset of U, or equivalently

the subset of edges with exactly one extremity in U. For an edge subset $F \subseteq E$ we also define $x(F) = \sum_{e \in F} x_e$, and if $F \subseteq \delta(\mathcal{D})$, $y(F) = \sum_{e \in F} y_e$. The vehicle-flow formulation of the MDVRP is as follows:

$$\min \quad \sum_{e \in E} t_e x_e + 2 \sum_{i \in \mathcal{D}, j \in \mathcal{C}} t_{ij} y_{ij} \tag{1}$$

$$x(\delta\{j\}) + 2y(\delta(\{j\}) \cap \delta(\mathcal{D})) = 2 \qquad j \in \mathcal{C} \tag{2}$$

$$x(\delta\{i\}) + 2y(\delta(\{i\}) \leq 2m_i \qquad i \in \mathcal{D} \tag{3}$$

$$x(\delta\{S\}) + 2y(\delta(S) \cap \delta(\mathcal{D})) \geq 2\max\{r(S), \rho(S)\} \qquad S \subseteq \mathcal{C} \tag{4}$$

$$x(\delta(S)) \geq 2[x(\delta(\{h\}) \cap \delta(\mathcal{D}')) + x(\delta(\{j\}) \cap \delta(\mathcal{D} \setminus \mathcal{D}'))] \qquad S \subseteq \mathcal{C}, h, j \in S,\ \mathcal{D}' \subset \mathcal{D} \tag{5}$$

$$y_e \in \{0,1\} \qquad e \in \delta(\mathcal{D}) \tag{6}$$

$$x_e \in \{0,1\} \qquad e \in E \tag{7}$$

The objective function aims to minimize the total traveling time. Constraints (2) are the degree constraints that impose each customer be visited exactly once. Constraints (3) is the fleet size constraint. They impose that at most m_i vehicles are used at each depot. Constraints (4) are the capacity and route length constraints. They impose that at least $\max\{r(S), \rho(S)\}$ vehicles are used to visit the customers of set S. Constraints (5) are the path constraints. They forbid routes to have the starting and ending points at two different depots. Finally, (6)-(7) impose that variables are indeed binary. This formulation is a particular case of the one introduced by Belenguer et al. [8] for the CLRP, with the only addition of the route length constraint represented by the constants ρ.

2.2 Set-partitioning formulation of the MDVRP

For each $i \in \mathcal{D}$ and $j \in \mathcal{C}$ we let y_{ij} be a binary variable equal to 1 iff customer j is served alone in a route departing from depot i, with cost equal to $2t_{ij}$. We now let Ω be the set of routes visiting at least two customers and respecting the capacity and route length constraints. For a depot $i \in \mathcal{D}$, we denote by Ω_i the subset of routes starting and ending i. For each $l \in \Omega$, we let θ_l be a binary variable equal to 1 iff route l is selected, and we denote by t_l its cost, which is equal to the sum of the traveling times along the edges used by l. For each customer $j \in \mathcal{C}$ and route $l \in \Omega$ we let a_j^l be the number of times that customer j is visited by l. For each depot subset $D \subseteq \mathcal{D}$ and customer subset $C \subseteq \mathcal{C}$ we let $y(D:C) = \sum_{i \in D} \sum_{j \in C} y_{ij}$. The set-partitioning formulation of the MDVRP is as follows:

$$\min \quad \sum_{l\in\Omega} t_l\theta_l + 2\sum_{i\in\mathcal{D},j\in\mathcal{C}} t_{ij}y_{ij} \tag{8}$$

$$\sum_{l\in\Omega} a_j^l\theta_l + y(\mathcal{D}:\{j\}) = 1 \qquad j\in\mathcal{D} \tag{9}$$

$$\sum_{l\in\Omega_i} \theta_l + y(\{i\}:\mathcal{C}) \le m_i \qquad i\in\mathcal{D} \tag{10}$$

$$y_{ij}\in\{0,1\} \qquad i\in\delta(\mathcal{D}) \tag{11}$$

$$\theta_l\in\{0,1\} \qquad l\in\Omega \tag{12}$$

Once again, the objective function aims to minimize the total traveling cost. Constraints (9) are the degree constraints that impose each customer be visited exactly once. Constraints (11)-(12) state the binary nature of the variables. Note that the capacity and the route length constraints are embedded into the definition of the route set Ω, and therefore do not need to be explicitely included in the formulation of the problem. For the same reason, the path constraints (5) are also not needed.

3 Valid inequalities

In this section we present the valid inequalities used to strengthen both formulations presented earlier. For the first families of inequalities, we refer to them as "weak" because their inclusion does not impose the addition of an extra resouce in the labeling algorithm. For the last five classes of inequalities introduced, we refer to them as "strong", because their addition imposes the use of additional resource during the labeling algorithm.

3.1 Weak valid inequalities

We call weak valid inequalities to all those inequalities that are valid for formulation (1) and that can be used in both formulations. These inequalities have the particularity that the contribution of their duals to the computation of the reduced costs of paths can be decomposed along the edges defining them, and therefore included in the pricing algorithm without compromising its performance. We consider some of the valid inequalities available in the CVRPSEP package [27], namely the framed capacity inequalities, strengthened comb inequalities, multistar inequalities and hypotour inequalities. We also use some of the inequalities introduced by Belenguer et al. [8], Contardo et al. [12], namely the y-capacity cuts, degree constraints and co-circuit inequalities using the separation algorithms of Contardo et al. [12].

3.2 Strong degree constraints

The strong degree constraints (SDC) were originally introduced by Contardo et al. [13] for the CLRP, and they are also valid for the CVRP and the MDVRP. Before presenting the inequality, let us define some notation. Given a customer $j\in\mathcal{C}$ and a route $l\in\Omega$, we let

ξ_j^l be a binary constant equal to 1 iff route l visits node j. For a given customer $j \in \mathcal{C}$ we define $\xi^\theta(j) = \sum_{l\in\Omega} \xi_j^l \theta_l$. The SDC associated to node j is

$$\xi^\theta(j) + y(\mathcal{D} : \{j\}) \geq 1. \tag{13}$$

Contardo et al. [13] proved that this constraint imposes partial elementarity on node j, this is, that no variable θ_l visiting node j twice or more will take a positive value in the solution of the linear relaxation of the set-partitioning formulation.

3.3 k-Cycle elimination constraints

We now introduce a new family of valid inequalities that can be seen as a weaker form of the strong degree constraints. Let $k \geq 1$ be an integer constant. Let $j \in \mathcal{C}$ be a customer and let $l \in \Omega$ be a route. Let us define ν_j^{kl} as the number of times that route l visits customer j with at least k nodes between two consecutive appearances of j in the route. In Figure 1 we illustrate by means of an example the behavior of the values ν_j^{kl} as k increases. The following k-cycle elimination constraint (k-CEC) is valid for the MDVRP:

$$\sum_{l\in\Omega} \nu_j^{kl}\theta_l + y(\mathcal{D} : \{j\}) \geq 1. \tag{14}$$

Because $\xi_j^l \leq \nu_j^{kl} \leq a_j^l$ it follows that the k-CEC is a strengthening of the degree constraint (9) but a weaker form of the SDC (13). The following proposition shows that the k-CEC for a given cycle length k and customer j is effective to forbid a cycle of length k or less from visiting customer j twice.

Theorem 3.1. *Suppose that a k-CEC has been added to problem (8)-(12) for customer j and cycle length k. All routes $l \in \Omega$ visiting node j twice or more and such that $\nu_j^{kl} < a_j^l$ will be non-basic in the linear relaxation of problem (8)-(12). In particular, no route visiting customer j two consecutive times with $k-1$ or less intermediate nodes will take a positive value.*

Proof If we consider the k-CEC and subtract from it the regular degree constraint (9) we obtain

$$\sum_{l\in\Omega} \left(\nu_j^{kl} - a_j^l\right) \theta_l \geq 0$$

The summation above only has interest for those l such that $\nu_j^{kl} < a_i^l$. In that case, it becomes

$$\sum_{l\in\Omega, \nu_j^{kl} < a_j^l} \left(\nu_j^{kl} - a_j^l\right) \theta_l \geq 0$$

This last summation implies that $\theta_l = 0$ for all l such that $\nu_j^{kl} < a_j^l$. □

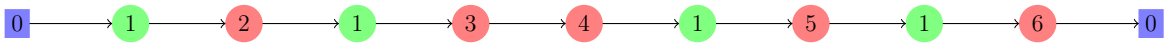

Figure 1: Route l with $a_1^l = 4$, $\nu_1^{1l} = 4$, $\nu_1^{2l} = 2$ and $\nu_1^{3l} = 1$

3.4 y-Strong capacity constraints

The y-strong capacity constraints (y-SCC) were also introduced by Contardo et al. [13]. Given a customer subset $S \subseteq \mathcal{C}$ and a route $l \in \Omega$, we let ξ_S^l be a binary constant equal to 1 iff route l visits at least one customer in S. We define $\xi^\theta(S) = \sum_{l\in\Omega} \xi_S^l \theta_l$. Also, let $r(S) = \lceil d(S)/Q \rceil$ be a lower bound on the number of vehicles that are needed to visit all customers in S, with $d(S) = \sum_{i\in S} d_i$. Let $S' \subseteq S$ be a subset satisfying $r(S \setminus S') = r(S)$. The following y-strong capacity constraints (y-SCC) is valid for the MDVRP:

$$\xi^\theta(S) + \sum_{i\in\mathcal{D}} \sum_{j\in S\setminus S'} y_{ij} \geq 1. \tag{15}$$

This inequality dominates the SCC introduced by Baldacci et al. [3] and the y-capacity constraints (y-CC) introduced by Belenguer et al. [8]. Its addition, however, imposes a modification in the pricing algorithm to properly handle the associated dual variable.

3.5 Strong framed capacity inequalities

The strong framed capacity inequalities (SFCI) are a lifted form of the original FCI and were introduced by Contardo et al. [13]. Given a customer subset $S \subseteq \mathcal{C}$ (the frame) and a partition of it $(S_k)_{k\in K}$, let $r(S, (S_k)_{k\in K})$ be the solution of the following bin-packing problem. For each set S_k with accumulated demand $d(S_k) \leq Q$, consider one object of size $d(S_k)$. For each set S_k with accumulated demand $d(S_k) \geq Q$ consider $n_k = \lfloor d(S_k)/Q \rfloor$ objects of size Q, plus at most one object of size $d(S) - Q \times n_k$ (this last object does not appear if $d(S)$ divides Q). Set the bins to have capacity Q. If $r(S, (S_k)_{k\in K}) > r(S)$ then the following inequality is valid for the MDVRP:

$$\xi^\theta(S) + \sum_{k\in K} \xi^\theta(S_k) + 2\sum_{i\in\mathcal{D}} \sum_{j\in S} y_{ij} \geq \sum_{k\in K} r(S_k) + r(S, (S_k)_{k\in K}) \tag{16}$$

Again, the addition of a SFCI requires a modification of the labeling algorithm during the recursion of the dynamic programming. Indeed, $|K| + 1$ additional resources are needed to properly handle the dual variable associated to a such constraint.

3.6 Subset-row inequalities

We consider two particular cases of subset-row inequalities [24]. Given a customer subset C of size n odd (we consider $n = 3, 5$), for every route $l \in \Omega$ we let n_C^l be the number of times that l visits the customers in C. The following inequality is a valid subset-row inequality for MDVRP:

$$\sum_{l\in\Omega} \lfloor n_C^l/2 \rfloor \theta_l \leq \lfloor n/2 \rfloor \tag{17}$$

The addition of a subset-row inequality of this form again forces the addition of an additional resource in the dynamic programming recursion.

3.7 Separation algorithms

For the weak constraints, we make use of the separation routines introduced in Lysgaard et al. [28] and Contardo et al. [12]. For the strong constraints, we use the same strategy as in Contardo et al. [13]. Constraints SDC and k-CEC are polynomial in number and can be easily separated by simple inspection. To find violated constraints y-SCC and SFCI, we verify for each weak constraint y-CC and FCI, if the strong version of such inequality is violated, and add it to the problem. Finally, for constraints SRI, we do the following. For $n = 3$, we check for every triplet (i, j, k) with $i < j < k$ if the corresponding SRI is violated and add it to the problem. For $n = 5$, this same procedure becomes impractical. Thus, we heuristically select the 30 customers with more appearances in the basic solutions of the current linear problem. We then consider all possible 5-tuples restricted to these 30 customers.

4 The exact method

In this section we describe the exact method used to solve the MDVRP to optimality. We first present a sketch of the algorithm. Then, we present the exact method decomposed into three main parts that are described in detail.

4.1 Sketch of the algorithm

In this section we sketch the exact algorithm used to solve the MDVRP. The algorithm is decomposed in three parts: variable fixing, column-and-cut generation, and column enumeration.

In the first part, the linear programming (LP) relaxation of the two-index vehicle-flow formulation is solved by means of the cutting planes method. Initially, the integrality constraints are dropped and we consider a restricted problem containing only a subset of constraints, namely (2)-(3). Constraints (4)-(5) are added as cutting planes as they are expontential in number. Other inequalities such as multistar inequalities, hypotour inequalities, framed capacity inequalities, strengthened comb inequalities, y-capacity cuts, degree constraints and co-circuit inequalities are also added as cutting planes. Note that the capacity and route length constraints (4) are relaxed by letting the right-hand side be equal to $r(S)$ (i.e., route length constraints are relaxed). The solution of the LP relaxation of formulation (1)-(6) is used to perform variable fixing based on the reduced costs of the edge variables $(x_e)_{e \in E}$.

In the second part of the algorithm, a reduced network is considered by a priori discarding all edges that were fixed to zero in the first stage. The set-partitioning formulation (8)-(12) is thus considered with the reduced set of edges. The LP relaxation of this problem is solved by colum-and-cut generation, in which the pricing subproblem corresponds to solve a 2-*cyc*-SPPRC for each depot. The problem is strengthened with the use of all inequalities introduced earlier. Initially, only some families of strong inequalities are added to the problem and we also let a hard limit on the maximum number of strong cuts. This hard limit is gradually increased, and each time it is reached we refer to it as a *major iteration*. To diversify the addition of cuts, each inequality is ranked according to its violation with respect to the right-hand side of the inequality.

In the third part of the algorithm, at the end of each major iteration, the resulting lower bound z_L is used, together with an upper bound z_U of the problem (that can be obtained with an heuristic method), to enumerate all elementary columns whose reduced costs are less than or equal to $z_U - z_L$. To bound the number of columns generated we set a hard limit on the maximum number of columns generated of 5 millions. If the enumeration procedure finishes with success, we consider a last major iteration in which the maximum number of separated inequalities is set to ∞ for all families in an attempt to strengthen the final lower bound and thus reduce the number of final columns in the problem.

4.2 First part: variable fixing

In the first stage of the algorithm, we consider formulation (1)-(7) strengthened with all families of valid inequalities already mentioned. The LP relaxation of this problem is solved by means of the cutting planes method. The relaxed constraints and the additional valid inequalities are sorted in the following order:

i. Subtour elimination constraints [16].

ii. y-Capacity cuts [8].

iii. Path constraints [8].

iv. Co-circuit constraints [8].

v. Framed capacity inequalities [1].

vi. Strengthened comb inequalities [28].

vii. y-Generalized large multistar inequalities [12].

viii. Multistar inequalities [28].

ix. Hypotour inequalities [28].

If a subtour elimination constraint is identified, the problem is immediately reoptimized. The same is applied when a y-capacity cut is identified. The remaining families of valid inequalities are inspected in order until detecting an inequality whose absolute violation is at least of 0.2 units, in which case the problem is immediately reoptimized.

When we are not able to identify any violated inequality, we stop and compute a lower bound z_{LB} of the problem and reduced costs $(\overline{t_e})_{e \in E}$ for the variables $(x_e)_{e \in E}$. This lower bound, together with a valid upper bound z_{UB} is used to fix to zero all variables x_e taking the value 0 in the current fractional solution and such that

$$\overline{t_e} + z_{LB} \geq z_{UB}. \tag{18}$$

Moreover, for instances with integer costs, we replace z_{UB} by the stronger quantity $z_{UB} - 0.99$. The quantity z_{UB} can be computed using any reasonable metaheuristic, such as the ones of Vidal et al. [31] or Cordeau and Maischberger [14]. Because implementing such algorithm is beyond the scope of this article, we use the best known upper bound as reported by Vidal et al. [31].

4.3 Second part: column-and-cut generation

In this section we describe the second part of the algorithm, which aims to solve the LP relaxation of problem (8)-(12) by column-and-cut generation, so as to provide a valid lower bound of the problem. We describe the pricing subproblem and the cut separation strategy.

4.3.1 Pricing subproblem

We consider the 2-*cyc*-SPPRC pricing subproblem. The 2-*cyc*-SPPRC can provide relatively strong bounds in very short computing times when compared to other relaxations of the ESPPRC. The addition of cuts y-CC, FCI and SCI does not affect the structure of the labeling algorithm producing tight bounds in very short times, that in many cases suffices to solve the problem. For hard instances, the bounds are strengthened with the use of the already mentioned strong inequalities. In particular, the addition of the SDC and k-CEC strengthens the poor bounds of the 2-*cyc*-SPPRC when the basic solutions contain too many cycles.

The 2-*cyc*-SPPRC is solved by means of bidirectional dynamic programming (BDP). In BDP, labels (or partial paths) are extended until reaching half of the capacity. To produce complete routes, partial paths are joined pairwise. The details of the complete dynamic programming method can be found in Contardo et al. [13]. Let us sketch the labeling algorithm and remark some of the main parts of it, namely the dominance and the fathoming rules used to discard non-promising labels.

The labeling algorithm starts with an empty label (or partial path) containing a depot node. For each label L we denote $v(L)$ its terminal node, $q(L)$ the current load of the vehicle, $t(L)$ the traveled time and $\overline{t}(L)$ its reduced cost. Each time a label L is extended to a customer node w, another label L' is created having $v(L') = w$, $q(L') = q(L) + d_w$ and $t(L') = t(L) + t_{v(L)w}$. The reduced cost is updated according to the reduced cost of the edge $\{v(L), w\}$ and the dual variables associated to the strong constraints. With each "strong" constraint $C \in \text{SDC} \cup y\text{-SCC}$ we associate a boolean resource equal to 1 iff the corresponding partial path visits at least one customer in the set defining the inequality. For a constraint $C \in \text{SFCI}$, we associate as many binary resources as the size of the partition plus one for the handle. We let $\mathcal{R}$ be the set of these boolean resources, and for each $\rho \in \mathcal{R}$ we denote σ_ρ the corresponding dual variable. Each of these boolean resources is updated from 0 to 1 and the associated dual variable is extracted from the reduced cost of that path when a partial path visits for the first time a node in the corresponding cutset defining the resource, which we denote by $CS(\rho)$. With each k-CEC C with dual variable $\sigma_C \geq 0$ associated with a customer j and a length k, we let κ be an integer resource initially set to k. Every time that j is visited, we verify if $\kappa \geq k$ in which case we subtract σ_C from the reduced cost of the path, and reset κ to zero (even if the dual variable is not subtracted, the resource is reset anyways). If the label was extended to another node different from j, κ is increased of one unit. We let $\mathcal{K}$ be the set of binding k-CEC in the current iteration. With each $C \in$ SRI with dual variable $\sigma_C \leq 0$, we also associate a binary resource r_C that now works as follows. The resource is initialized to zero, and whenever the label reaches a customer in the SRI, it is updated to one. The following time that the label visits a customer in the SRI, we subtract σ_C from the reduced cost of the label and reset the resource to zero. We let $\mathcal{S}$ be the set of the resources associated to SRI inequalities.

We use the following dominance rule to discard labels by doing pairwise comparisons. Let L, L' be two labels (or partial paths), starting at the same depot (in the case of the MDVRP), ending at nodes $v(L), v(L')$, with loads $q(L), q(L')$, accumulated traveling times $t(L), t(L')$ and reduced costs $\overline{t}(L), \overline{t}(L')$. We say that L dominates L' (and denote it as $L \succ L'$) if

i. $v(L) = v(L')$.

ii. $q(L) \leq q(L')$.

iii. $t(L) \leq t(L')$.

iv. $\overline{t}(L) - \sum_{C \in \mathcal{S}, r_C(L) > r_C(L')} \sigma_C \leq \overline{t}(L') - \sum_{\rho \in \mathcal{R}, \rho(L) > \rho(L')} \sigma_\rho - \sum_{C \in \mathcal{K}, \kappa(L) < \kappa(L')} \sigma_C$.

This dominance rule cannot be solely applied to discard labels in a 2-*cyc*-SPPRC algorithm. Indeed, it does not consider that labels may or may not be back to the predecessor node, which may result in dominating a label that can be extended to more nodes than the dominant. To solve that issue and provide an efficient implementation of the 2-*cyc*-SPPRC, Kohl [25] and Larsen [26] define the concept of *strongly dominant* labels, *semi-strongly dominant* labels and *weakly dominant* labels. A *strongly dominant* label is a label that is not dominated by any other label and that cannot be extended to its predecessor node due to the resource constraints. A *semi-strongly dominant* label is a label that is not dominated by any other label but is not strongly dominant. Finally, a *weakly dominant* label is a label that is dominated only by semi-strongly dominant labels, all of which share the same predecessor node, which is also different from the predecessor node of the label. Any label that is not strongly dominant, semi-strongly dominant or weakly dominant can be discarded.

This dominance rule can be sharpened for the case of the CVRP by realizing that condition (iii) can be omitted. Note that this pricing subproblem is exact and can be time consuming in the presence of too many resources. Hence, we always perform the following heuristic pricing before applying the exact procedure. First, we limit the percentage of edges to κ. To select the edges, we perform the underestimation of the binding strong constraints (to be explained later in the text) and rank the edges according to the resulting lower bounds on the edge reduced costs. Second, the SDC and k-CEC are used to impose the asociated strucral constraints (let us remark that these two families of valid inequalities cannot be used in an exact pricing to impose structural constraints -i.e. to forbid cycles- in which case one should resort to the classical dominance rule for elementarity and k-cycle elimination constraints). The labeling algorithm is run on the modified network with the additional structural constraints for four diferent values of κ before resorting to the exact pricing method, namely we use $\kappa = 0.2, 0.4, 0.7, 1.0$.

Now, let us present the fathoming rule used to discard a label L based on its reduced cost $\overline{t}(L)$ and a completion bound $LB(L)$, computed as follows. Let $\mathcal{S}$ be the set of all subset-row inequalities (as described in 3.6) added to the problem, and let $\mathcal{T} \subseteq \mathcal{S}$ be a subset of it. Let $C \in \mathcal{T}$ be a SRI with associated dual variable $\sigma \leq 0$. Let us define the *underestimation of* C as the following procedure: To each edge e with both endpoints in C we subtract $\sigma/2$ from the reduced cost of the edge and discard the resource associated to such constraint. Because σ is negative, this procedure has as a consequence that the modified reduced cost of a path is indeed a lower bound of the actual reduced cost. In a similar way, Contardo et al. [13]

showed that y-SCC, SFCI and SDC can also be underestimated. The underestimation of a k-CEC is done in an analogous manner as for the SDC. Indeed, if C is a k-CEC associated with a cycle length k and a customer j with dual variable $\sigma_C \geq 0$, we subtract $\sigma_C/2$ from the reduced cost of every edge having j as one of its endpoints. Now, let us consider the following auxiliar pricing problem. From all the constraints "strong", let us keep the resources associated to those with the largest duals, and perform an underestimation of the remaining ones. Let us denote by $\mathcal{R}_U$ the resources in $\mathcal{R}$ associated to the constraints that have been underestimated, $\mathcal{K}_U \subseteq \mathcal{K}$ be the subset of k-CEC that have been underesitimated and $\mathcal{T}_U \subseteq \mathcal{T}$ be the subset-row inequalities that have been underestimated. Solve the resulting 2-*cyc*-SPPRC with now less resources, and let $f(q,t,j)$ be the minimum reduced cost among the partial paths having j as endpoint, with load $l \leq Q - q + d_j$ and arrival time to j less or equal than $T - t$. Similarly, let $\pi(q,t,j)$ be the predecessor of j in such partial path. Now, let $g(q,t,j)$ be the second lowest reduced cost not having $\pi(q,t,j)$ as predecessor node. For each label L, let $u_1(L), u_2(L), u_3(L), u_4(L), u_5(L)$ and $u_6(L)$ be defined as follows:

$$u_1(L) = \sum_{\{\rho:\rho\in\mathcal{R}\setminus\mathcal{R}_U, v(L)\in CS(\rho)\}} \sigma_\rho \tag{19}$$

$$u_2(L) = \sum_{\{\rho:\rho\in\mathcal{R}_U, v(L)\in CS(\rho)\}} \sigma_\rho \tag{20}$$

$$u_3(L) = \sum_{\{C\in\mathcal{K}\setminus\mathcal{K}_U: C \text{ is associated to node } v(L)\}} \sigma_C \tag{21}$$

$$u_4(L) = \sum_{\{C\in\mathcal{K}_U: C \text{ is associated to node } v(L)\}} \sigma_C \tag{22}$$

$$u_5(L) = \sum_{\{C:C\in\mathcal{T}\setminus\mathcal{T}_U, v(L)\in C, r_C(L)=0\}} \sigma_C \tag{23}$$

$$u_6(L) = \sum_{\{C:C\in\mathcal{T}_U, v(L)\in C, r_C(L)=0\}} \sigma_C. \tag{24}$$

For a given label (or partial path) L having v as endpoint, p as predecessor node, with accumulated capacity of q units and arrival time t, a lower bound on the possible extensions of L can be computed as

$$LB(L) = \overline{t}(L) + u_1(L) + u_3(L) + u_5(L) + \frac{u_2(L)+u_4(L)+u_6(L)}{2} + \begin{cases} f(q,t,v) & \text{if } p \neq \pi(q,t,v) \\ g(q,t,v) & \text{otherwise} \end{cases} \tag{25}$$

This quantity effectively defines a lower bound on the reduced cost potentially achieved by the extensions of label L to a complete path, and can be used to discard a label when the completion bound $LB(L)$ is larger than or equal to zero. Note that because the domain of functions f, π and g can be extremely large (even infinity if travel times are arbitrary real numbers), we define them on a limited number of buckets of load and time. The finer the granularity used in these buckets, the more accurate that the resulting bounds will be, but also more memory space will be needed to store such quantities.

At this stage of the algorithm, we keep resources for at most 20% of the binding strong constraints, and underestimate the remaining ones. In addition, we use $\mathcal{T} = \mathcal{S}$. Note that

different subsets $\mathcal{T}$ can be used to derive different completion bounds. After some preliminary tests, we did not find a clear dominance criterion between different choices of $\mathcal{T}$ (we used $\mathcal{T} = \{\emptyset, \mathcal{S}\}$), which may be an avenue of future research.

4.3.2 Cut separation strategy

The strategy for adding cuts is as follows: first of all, we only add cuts when the current restricted master problem is proven to be feasible (which may not be true at the beginning of the column generation, or right after the addition of new valid inequalities). We always try to add weak inequalities first. When we cannot identify any violated weak inequality, we try to add columns of negative reduced cost (for which we use first the heuristic pricing and then the exact pricing algorithm). If we cannot find columns of negative reduced cost, we separate and add strong valid inequalities. The strategy for adding strong cuts is as follows. At the first major iteration, only k-CEC are added, for $3 \leq k \leq 7$. At the second major iteration, we also include cuts y-SCC, SFCI and SRI for $n = 3$. In the subsequent iterations, we also allow the addition of cuts SDC and SRI for $n = 5$. The maximum number of strong cuts other than k-CEC after the first major iteration is 100, 200 and 400, respectively.

4.4 Third part: column enumeration

At the end of a major iteration, we use a dynamic programming algorithm similar to the one used in the 2-*cyc*-SPPRC to enumerate all columns that may potentially lead to an improvement of the upper bound. The main difference is the use of a stronger dominance criterion, and imposing elementarity to forbid the generation of routes containing cycles. In the dynamic programming algorithm, a label L represents a partial path starting from a depot, and is composed of the following quantities: the terminal node $v(L)$, the predecessor label $pred(L)$, the reduced cost $\overline{t}(L)$, the customers visited $V(L)$, the size of that set $|V(L)|$, the load $q(L)$ and the accumulated traveling time $t(L)$. These quantities are computed accordingly after every extension of a label to its possible successors. Now, a modified dominance rule is used to discard labels. Indeed, because we enumerate all columns that may potentially appear in the optimal solution, dominance must be done using a much stronger criterion. Notably, we say that a label L dominates a label L' if all of the following conditions hold:

i. $v(L) = v(L')$.

ii. $q(L) = q(L')$.

iii. $|V(L)| = |V(L')|$.

iv. $V(L) = V(L')$.

v. $t(L) \leq t(L')$.

Note that although conditions (ii)-(iii) are implied by (iv), they are used to rapidly discard the dominance of one label with respect to another when any of these conditions is not satisfied. Note also that because we want to enumerate all feasible routes, only

elementary labels are generated and, moreover, label joints are done only using pairs of labels that are customer-disjoint, this is, that only share the depot and terminal nodes.

This dominance rule, because it is much stronger than the classical dominance used for the 2-*cyc*-SPPRC, may not be sufficient to bound the computational complexity of the enumeration procedure. Therefore, we complement it with a fathoming rule which is based on the solution of the 2-*cyc*-SPPRC with all strong resources, after which a completion bound can be obtained using expression (25).

Similar enumeration procedures have been used in [3, 2, 4, 6, 7, 13] and for more details on the enumeration procedures we refer to these articles.

5 Computational experiments

We have conducted a series of experiments on several sets of instances from the literature for both problems considered in this study. For the CVRP, we consider six classes of instances, namely classes A, B, E, F, M and P. Classes A, B and P were proposed by Augerat [1]. Class E was proposed by Christofides and Eilon [9]. Class F was proposed by Fisher [18]. Finally, class M was proposed by Christofides et al. [10]. All these instances can be found in the website `http://www.branchandcut.org`. For the MDVRP, we consider 18 instances from the dataset proposed by Cordeau et al. [15]. The complete dataset can be found in `http://neo.lcc.uma.es/radi-aeb/WebVRP`. The algorithm has been coded in C++, compiled using the Intel Compiler v11.0 and run on an Intel Xeon E5462 2.8 GHz with 16GB of RAM.

In Tables 1-6 we present the computational results obtained with the proposed algorithm for both problems considered in this study. In these tables, column labeled "UB" represents the best known solution of the problem, as reported by previous methods. Column labeled "z^*" represents the best solution found by the proposed method. Under column labeled "Cutting Planes" we report the lower bound, optimality gap (in %) and CPU time taken to solve the LP relaxation of problem (1)-(6). Under column labeled "Column and Cut Generation" we report the results obtained by the column-and-cut generation method. Column labeled "LB_r" represents the lower bound achieved at the end of the LP relaxation. Column labeled "gap_r" represents the gap at the end of the LP relaxation. Column labeled "t_r" represents the time spent at solving the root node. Column labeled "$|\mathcal{R}|$ gen" represents the number of routes generated by the eumeration algorithm. Column labeled "$|\mathcal{R}|$ fin" represents the final number of column kept after the strengthening of the lower bound and subsequent variable fixing. Column labeled "LB_f" represents the final lower bound. Column labeled "gap_f" represents the final gap. Column labeled "t_{cpx}" represents the time spent by CPLEX to solve the final integer program. Finally, column labeled "t_{tot}" represents the total CPU time spent. We use bold characters for the instances solved to optimality. As shown in these tables, our exact method is effective for solving all but three instances for the CVRP, namely instances M-n200-k16, M-n200-k17 and F-n135-k7. The first two instances remain open, while the third instance has been already solved by Fukasawa et al. [19] using a branch-and-cut algorithm. This result confirms the statement already claimed by previous authors that problems with loose capacity constraints may be difficult to solve with column generation algorithms, as is the case with problem F-n135-k7. On the other

hand, our algorithm has solved instance M-n151-k12 for the first time in less than 6 hours, which shows that our method takes advantage of the enhacements introduced with respect to previous column generation methods. For the MDVRP, our algorithm is able to solve all 18 instances considered in our study, including 11 instances that have been solved for the first time. Moreover, our algorithm has improved the best known solution of instance pr07 which is also proven to be optimal (see the new solution in Figure 2).

Instance	UB	z^*	Cutting Planes			Column and Cut Generation								
			LB	gap	T	LB_r	gap_r	t_r	$\|\mathcal{R}\|$ gen	$\|\mathcal{R}\|$ fin	LB_f	gap_f	t_{cpx}	t_{tot}
A-n37-k5	669	**669**	663.85	0.77	1.72	669.00	0.00	6.23	192	0	669	0.00	0.00	7.95
A-n37-k6	949	**949**	924.18	2.62	3.92	945.96	0.32	16.50	1951	213	949	0.00	0.14	20.56
A-n38-k5	730	**730**	716.34	1.87	1.99	728.80	0.16	11.41	2552	68	730	0.00	0.06	13.46
A-n39-k5	822	**822**	808.57	1.63	5.44	822.00	0.00	14.01	8722	0	822	0.00	0.00	19.45
A-n39-k6	831	**831**	814.84	1.95	4.21	831.00	0.00	8.62	793	0	831	0.00	0.00	12.83
A-n44-k6	937	**937**	920.34	1.78	7.77	937.00	0.00	3.73	137	0	937	0.00	0.00	11.50
A-n45-k6	944	**944**	928.88	1.60	4.68	944.00	0.00	25.71	134	0	944	0.00	0.00	30.39
A-n45-k7	1146	**1146**	1112.11	2.96	12.96	1146.00	0.00	14.71	879	0	1146	0.00	0.00	27.67
A-n46-k7	914	**914**	909.33	0.51	3.45	914.00	0.00	3.69	0	0	914	0.00	0.00	7.14
A-n48-k7	1073	**1073**	1052.51	1.91	10.17	1073.00	0.00	14.14	74	0	1073	0.00	0.00	24.31
A-n53-k7	1010	**1010**	996.80	1.31	5.04	1010.00	0.00	24.97	4200	0	1010	0.00	0.00	30.01
A-n54-k7	1167	**1167**	1132.98	2.92	11.1	1167.00	0.00	84.31	169116	0	1167	0.00	0.00	95.41
A-n55-k9	1073	**1073**	1056.40	1.55	4.43	1073.00	0.00	13.85	1389	0	1073	0.00	0.00	18.28
A-n60-k9	1354	**1354**	1316.98	2.73	21.39	1354.00	0.00	37.15	40136	0	1354	0.00	0.00	58.54
A-n61-k9	1034	**1034**	1007.76	2.54	8.41	1032.60	0.14	52.32	17544	169	1034	0.00	0.30	61.03
A-n62-k8	1288	**1288**	1249.36	3.00	17.09	1288.00	0.00	87.55	65200	0	1288	0.00	0.00	104.64
A-n63-k9	1616	**1616**	1576.76	2.43	18.36	1616.00	0.00	59.15	7997	0	1616	0.00	0.00	77.51
A-n63-k10	1314	**1314**	1265.05	3.73	20.84	1309.63	0.33	46.56	18528	1292	1314	0.00	6.26	73.66
A-n64-k9	1401	**1401**	1349.32	3.69	20.51	1396.18	0.34	170.21	80659	1777	1401	0.00	33.59	224.31
A-n65-k9	1174	**1174**	1153.41	1.75	13.91	1174.00	0.00	33.40	4033	0	1174	0.00	0.00	47.31
A-n69-k9	1159	**1159**	1112.56	4.01	19.28	1157.39	0.14	70.32	93739	150	1159	0.00	0.29	89.89
A-n80-k10	1763	**1763**	1707.14	3.17	55.63	1763.00	0.00	192.26	28161	0	1763	0.00	0.00	247.89
Average				2.25	10.32		0.07	38.03				0.00	1.94	50.28

Table 1: Detailed results on class A for the CVRP

Instance	UB	z^*	Cutting Planes			Column and Cut Generation								
			LB	gap	T	LB_r	gap_r	t_r	$\|\mathcal{R}\|$ gen	$\|\mathcal{R}\|$ fin	LB_f	gap_f	t_{cpx}	t_{tot}
B-n38-k6	805	**805**	800.188	0.60	1.3	805.00	0.00	3.09	0	0	805	0.00	0.00	4.39
B-n39-k5	549	**549**	548.5	0.09	0.18	549.00	0.00	0.00	0	0	549	0.00	0.00	0.18
B-n41-k6	829	**829**	826.167	0.34	1.28	829.00	0.00	3.71	0	0	829	0.00	0.00	4.99
B-n43-k6	742	**742**	733.5	1.15	1.84	740.10	0.26	22.44	23396	466	742	0.00	0.29	24.57
B-n44-k7	909	**909**	862.5	5.12	0.74	909.00	0.00	5.57	0	0	909	0.00	0.00	6.31
B-n45-k5	751	**751**	747.367	0.48	2.54	751.00	0.00	18.44	0	0	751	0.00	0.00	20.98
B-n45-k6	678	**678**	673.566	0.65	6.33	678.00	0.00	14.25	0	0	678	0.00	0.00	20.58
B-n50-k7	741	**741**	729.6	1.54	0.86	741.00	0.00	2.43	0	0	741	0.00	0.00	3.29
B-n50-k8	1312	**1312**	1280.52	2.40	9.2	1309.94	0.16	132.39	651978	1471	1312	0.00	2.74	144.33
B-n51-k7	1032	**1032**	1024.75	0.70	2.41	1032.00	0.00	27.62	91040	0	1032	0.00	0.00	30.03
B-n52-k7	747	**747**	745.833	0.16	1.9	747.00	0.00	1.60	0	0	747	0.00	0.00	3.50
B-n56-k7	707	**707**	703.816	0.45	4.28	705.00	0.28	22.18	433	433	707	0.00	0.10	26.56
B-n57-k7	1153	**1153**	1149.07	0.34	7.39	1153.00	0.00	115.97	0	0	1153	0.00	0.00	123.36
B-n57-k9	1598	**1598**	1588.11	0.62	8.47	1598.00	0.00	28.84	335	0	1598	0.00	0.00	37.31
B-n63-k10	1496	**1496**	1479.56	1.10	7.34	1496.00	0.00	81.69	124871	0	1496	0.00	0.00	89.03
B-n64-k9	861	**861**	859.192	0.21	6.62	861.00	0.00	37.86	0	0	861	0.00	0.00	44.48
B-n66-k9	1316	**1316**	1297.99	1.37	18.22	1316.00	0.00	184.29	1407605	0	1316	0.00	0.00	202.51
B-n67-k10	1032	**1032**	1023.96	0.78	5.93	1032.00	0.00	54.92	10562	0	1032	0.00	0.00	60.85
B-n68-k9	1272	**1272**	1257.17	1.17	16.58	1268.03	0.31	507.01	3278498	14970	1272	0.00	239.26	762.85
B-n78-k10	1221	**1221**	1204.3	1.37	25.45	1221.00	0.00	144.18	16087	0	1221	0.00	0.00	169.63
Average				1.03	6.44		0.05	70.42				0.00	12.12	88.99

Table 2: Detailed results on class B for the CVRP

Instance	UB	z^*	Cutting Planes			Column and Cut Generation								
			LB	gap	T	LB_r	gap_r	t_r	$\|\mathcal{R}\|$ gen	$\|\mathcal{R}\|$ fin	LB_f	gap_f	t_{cpx}	t_{tot}
P-n40-k5	458	**458**	456.70	0.28	3.47	458	0.00	0.64	0	0	458	0.00	0.00	4.11
P-n45-k5	510	**510**	504.08	1.16	2.51	510	0.00	9.61	2832	0	510	0.00	0.00	12.12
P-n50-k7	554	**554**	539.46	2.63	12.16	554	0.00	7.09	822	0	554	0.00	0.00	19.25
P-n50-k8	631	**631**	555.00	12.04	0.05	626.226	0.76	38.35	21222	803	631	0.00	3.10	41.50
P-n50-k10	696	**696**	666.86	4.19	14.24	696	0.00	4.88	1514	0	696	0.00	0.00	19.12
P-n51-k10	741	**741**	714.39	3.59	12.31	741	0.00	7.64	798	0	741	0.00	0.00	19.95
P-n55-k7	568	**568**	548.27	3.47	10.00	565.667	0.41	35.96	50768	435	568	0.00	1.57	47.53
P-n55-k8	588	**588**	568.90	3.25	3.81	586.141	0.32	13.11	17054	175	588	0.00	0.49	17.41
P-n55-k10	694	**694**	660.87	4.77	16.36	689.234	0.69	16.31	25833	1386	694	0.00	6.26	38.93
P-n55-k15	989	**989**	906.25	8.37	66.99	984.638	0.44	17.56	2943	305	989	0.00	1.20	85.75
P-n60-k10	744	**744**	717.08	3.62	17.64	744	0.00	10.13	1528	0	744	0.00	0.00	27.77
P-n60-k15	968	**968**	928.64	4.07	22.76	968	0.00	7.63	543	0	968	0.00	0.00	30.39
P-n65-k10	792	**792**	766.18	3.26	30.50	792	0.00	11.74	976	0	792	0.00	0.00	42.24
P-n70-k10	827	**827**	794.19	3.97	32.10	823.171	0.46	66.45	211043	1837	827	0.00	34.06	132.61
P-n76-k4	593	**593**	588.57	0.75	13.48	593	0.00	349.24	1155576	0	593	0.00	0.00	362.72
P-n76-k5	627	**627**	616.71	1.64	21.87	627	0.00	1309.02	890384	0	627	0.00	0.00	1330.89
P-n101-k4	681	**681**	678.40	0.38	24.77	681	0.00	5768.49	359788	0	681	0.00	0.00	5793.26
Average				3.61	17.94		0.18	451.40				0.00	2.75	472.09

Table 3: Detailed results on class P for the CVRP

Instance	UB	z^*	Cutting Planes			Column and Cut Generation								
			LB	gap	T	LB_r	gap_r	t_r	$\|\mathcal{R}\|$ gen	$\|\mathcal{R}\|$ fin	LB_f	gap_f	t_{cpx}	t_{tot}
E-n51-k5	521	**521**	518.231	0.53	5.42	521.00	0.00	8.31	476	0	521	0.00	0.00	13.73
E-n76-k7	682	**682**	665.471	2.42	29.41	682.00	0.00	349.28	475630	0	682	0.00	0.00	378.69
E-n76-k8	735	**735**	716.746	2.48	31.55	735.00	0.00	140.21	459826	0	735	0.00	0.00	171.76
E-n76-k10	830	**830**	797.688	3.89	46.05	826.55	0.42	102.06	620157	2061	830	0.00	35.77	183.88
E-n76-k14	1021	**1021**	968.041	5.19	64.88	1014.76	0.61	41.51	82822	3976	1021	0.00	34.91	141.30
E-n101-k8	815	**815**	800.859	1.74	45.36	815.00	0.00	1046.82	1064808	0	815	0.00	0.00	1092.18
E-n101-k14	1067	**1067**	1024.47	3.99	170.35	1063.45	0.33	404.31	100096	6400	1067	0.00	87.81	662.47
M-n101-k10	820	**820**	820	0.00	13.16	820.00	0.00	0.00	0	0	820	0.00	0.00	13.16
M-n121-k7	1034	**1034**	1015.64	1.78	122.42	1034.00	0.00	5590.90	224325	0	1034	0.00	0.00	5713.32
M-n151-k12	1015	**1015**†	973.245	4.11	373.6	1012.48	0.25	18669.20	3975788	13117	1015	0.00	656.32	19699.12
M-n200-k16	1278	1278‡	1203.42	5.84	1211.72	1263.00	1.17	264377.00	∞	∞	1263.00	1.17	0.00	265588.72
M-n200-k17	1275	1275‡	1190.62	6.62	418.55	1265.08	0.78	33932.00	∞	∞	1265.08	0.78	0.00	34350.55
Average				3.22	211.04		0.30	27055.13				0.16	67.90	27334.07

† Optimality proven for the first time

‡ Optimality not proven

Table 4: Detailed results on classes E-M for the CVRP

Instance	UB	z^*	Cutting Planes			Column and Cut Generation								
			LB	gap	T	LB_r	gap_r	t_r	$\|\mathcal{R}\|$ gen	$\|\mathcal{R}\|$ fin	LB_f	gap_f	t_{cpx}	t_{tot}
F-n45-k4	724	**724**	724	0.00	0.70	724	0.00	0.00	0	0	724	0.00	0.00	0.70
F-n72-k4	237	**237**	237	0.00	4.00	237	0.00	0.00	0	0	237	0.00	0.00	4.00
F-n135-k7	1162	1162‡	1159.58	0.21	93.39	1159.85	0.19	> 5 days	∞	∞	1159.85	0.19	0.00	> 5 days
Average				0.07	32.70		0.06					0.06	0.00	

‡ Optimality not proven

Table 5: Detailed results on class F for the CVRP

Instance	UB	z^*	Cutting Planes			Column and Cut Generation								
			LB	gap	T	LB	gap	T LB	$\|\mathcal{R}\|$ gen	$\|\mathcal{R}\|$ fin	LB_f	gap_f	T CPX	T tot
p01	576.87	**576.87**	543.58	5.77	11.97	576.87	0.00	13.83	148	0	576.87	0.00	0.00	25.80
p02	473.53	**473.53**	452.73	4.39	4.34	473.25	0.06	52.47	5526	124	473.53	0.00	0.13	56.94
p03	641.19	**641.19**	614.11	4.22	36.00	640.22	0.15	74.96	12182	271	641.19	0.00	0.21	111.17
p04	1001.04	**1001.04**	950.22	5.08	151.42	997.38	0.37	458.87	2340469	13242	1001.04	0.00	183.30	793.59
p05	750.03	**750.03**†	732.27	2.37	32.89	748.30	0.23	9718.07	3212798	12033	750.03	0.00	1431.53	11182.49
p06	876.50	**876.50**	831.83	5.10	123.96	875.28	0.14	240.73	257010	644	876.50	0.00	0.73	365.42
p07	881.97	**881.97**	832.50	5.61	99.58	881.81	0.02	289.21	313124	202	881.97	0.00	0.22	389.01
p12	1318.95	**1318.95**	1273.06	3.48	19.16	1313.80	0.39	1017.10	26393	1196	1318.95	0.00	1.45	1037.71
p13	1318.95	**1318.95**†	1273.06	3.48	19.16	1318.95	0.00	94.15	0	0	1318.95	0.00	0.00	113.31
p14	1360.12	**1360.12**†	1273.06	6.40	18.53	1360.12	0.00	47.36	53	0	1360.12	0.00	0.00	65.89
p15	2505.42	**2505.42**†	2380.36	4.99	239.83	2505.42	0.00	13154.20	39494	232	2505.42	0.00	0.24	13394.27
p16	2572.23	**2572.23**†	2380.36	7.46	238.05	2572.23	0.00	966.08	3734	0	2572.23	0.00	0.00	1204.13
p17	2709.09	**2709.09**†	2380.36	12.13	240.42	2709.09	0.00	939.68	255	0	2709.09	0.00	0.00	1180.10
p18	3702.85	**3702.85**†	3487.65	5.81	859.03	3699.78	0.08	69275.90	1196507	2213	3702.85	0.00	21.13	70156.06
p19	3827.06	**3827.06**†	3487.65	8.87	878.61	3827.06	0.00	5150.45	17501	0	3827.06	0.00	0.00	6029.06
p20	4058.07	**4058.07**†	3434.47	15.37	281.39	4058.07	0.00	4637.16	0	0	4058.07	0.00	0.00	4918.55
pr01	861.32	**861.32**†	849.17	1.41	1.12	861.32	0.00	17.10	14	0	861.32	0.00	0.00	18.22
pr07	1089.56	**1077.33**†	1057.49	1.84	5.05	1077.33	0.00	2612.95	1100847	0	1077.33	0.00	0.00	2618.00
Average				5.77	181.14		0.08	6042.24				0.00	91.05	6314.43

† Optimality proven for the first time

Table 6: Detailed results on selected MDVRP instances

In Tables 7-8 we present comparative results against state-of-the-art exact methods for the two classes of problems considered in this study. For the CVRP, we consider methods FLLPRUW [19], BCM [3], BBMR [4] and BMR [6]. For the proposed method, we report the results under column PM. For each of the methods, we report the number of instances solved to optimality (*opt*), the average gap (in %) at the LP relaxation of the set-partitioning formulation and the average CPU time on the instances that are solved by all methods. For the MDVRP, we compare our results against method BM [2] which is also based on a set-partitioning formulation of the problem. We provide a detailed comparison including all instances considered in our study. However, the average gap and CPU times reported includes only the instances that are considered by both methods. Moreover, the average CPU times are computed by taking into account only the instances solved to optimality by both methods. According to SPEC (`http://www.spec.org/benchmarks.html`), the machine used by us is about 4 times faster than the Pentium 4 2.6 GHz used by FLLPRUW and the Pentium 4 2.4 GHz used by method BCM, 2 times faster than the AMD Athlon X2 4200+ 2.6GHz used by method BM, 50% faster than the Intel Core 2 Duo P8400 2.26GHz used by BBMR and 25% faster than the Intel Xeon X7350 2.93 GHz used by method BMR. As the results show, our method is competitive against other exact methods for the CVRP, being able to obtain the tightest average gaps at the LP relaxation for all classes of instances. In terms of computing time, our method is comparable to methods BCM, BMR and BBMR. However, it is more robust than these three methods as it solves more problems and achieves better bounds. When compared to method FLLPRUW, our method in general is much faster in most instances, however it fails to solve problem F-n135-k7 that is solved by the authors using a branch-and-cut method. For the MDVRP, our method is more robust than method BM. In fact, it solves all instances also solved by method BM plus two instances that their algorithm fails to solve. For the commonly solved instances, the CPU times are comparable but our method in general achieves tighter lower bounds at the LP relaxation.

Class	NP	FLLPRUW			BCM			BBMR			BMR			PM		
		opt	gap	t	opt	gap	t	opt	gap	t	opt	gap	t	opt	gap	t
A	22	**22**	0.81	1961.18	**22**	0.20	118.07	**22**	0.30	22.00	**22**	0.13	30.27	**22**	**0.07**	59.26
B	20	**20**	0.47	4763.00	**20**	0.16	417.17	**20**	0.10	66.00	**20**	0.06	66.75	**20**	**0.05**	88.99
E-M	12	9	1.19	42614.50	8	0.69	1025.13	9	0.50	249.00	9	0.49	268.13	**10**	**0.30**	909.79
F	3	**3**	0.14	64.50							2	0.11	163.50	2	**0.06**	2.35
P	17	**17**	1.07	3572.53	15	0.39	272.07	**17**	0.20	54.00	**17**	0.23	39.80	**17**	**0.18**	60.09
Total	74	**71**			65			68			70			**71**		
Average			0.84	8198.62		0.36	357.28		0.28	70.86		0.21	72.97		**0.13**	173.28

Table 7: Summary results for the CVRP

6 Conluding remarks

In this article we have presented a new exact method for a multi-depot vehicle routing problem under capacity and route length constraints. The problem is modeled using ad-hoc vehicle-flow and set-partitioning formulations, and solved the first by the cutting planes method and the second by column-and-cut generation. Several families of valid inequalities are used to strengthen the lower bounds achieved by both formulations, including a new

Instance	BM		PM	
	gap	t	gap	t
p01	**0.00**	10.90	**0.00**	25.80
p02	0.30	54.40	**0.06**	56.94
p03	**0.10**	92.10	0.15	111.17
p04	0.70	5106.90	**0.37**	793.59
p05	2.50		**0.23**	11182.49
p06	0.40	104.30	**0.14**	365.42
p07	0.50	294.40	**0.02**	389.01
p12	1.40	463.70	**0.39**	1037.71
p13			**0.00**	113.31
p14			**0.00**	65.89
p15	0.80		**0.00**	13394.27
p16			**0.00**	1204.13
p17			**0.00**	1180.10
p18			**0.08**	70156.06
p19			**0.00**	6029.06
p20			**0.00**	4918.55
pr01			**0.00**	18.22
pr07			**0.00**	2618.00
Average	0.74	875.24	**0.15**	397.09
Opt	7/18		**18/18**	

Table 8: Comparison for the MDVRP

Depot	Load	Time	Route
73	193	242.15	27 60 70 30 42 19 9 11 6 33 65 10 31 20 18 71 36
74	193	220.025	49 61 37 58 7 43 26 23 1 64 47 12 68
75	196	125.659	8 45 15 16 3 56 54 22 34 44 62 69
76	196	288.487	13 59 51 17 41 50 57 24 63 5 72 53 46 28 29 52
77	74	90.333	14 67 40 4 55 21 48 66
78	96	110.676	25 32 39 38 2 35

Figure 2: New solution of value 1077.33 for instance pr07

family of valid inequalities that is shown to forbid cycles of an arbitrary length. As a result, our method is able to produce the tightest lower bounds for two classes of problems, the MDVRP and the CVRP, on the instances considered in our study when compared to state-of-the-art methods, and is able to solve 12 open instances, one for the CVRP and 11 for the MDVRP. We can identify several avenues of future research. For instance, improving the cutting planes would result in an even faster and more robust pricing algorithm, which would be of great use for solving poorly constrained instances. Another avenue of further research would be the generalization of the k-CEC and SDC cuts to forbid other types of cycles, in the spirit of the ng-routes relaxation. Another possible avenue of further research would be to investigate the impact in the computing times of the ng-routes relaxation when combined with the k-CEC and SDC to produce near-elementary bounds. Finally, let us remark that the SDC and k-CEC are two families of valid inequalities that can be adapted to several other classes of vehicle routing problems, like vehicle routing problems with time windows, multiple-echelon vehicle routing problems or multiple-period vehicle routing problems. Thus, future exact algorithms for these classes of problems should consider the inclusion of these cuts.

References

[1] P. Augerat. *Approche polyhédrale du problème de tournées de véhicules*. PhD thesis, Institut National Polytechnique de Grenoble, France, 1995.

[2] R. Baldacci and A. Mingozzi. A unified exact method for solving different classes of vehicle routing problems. *Mathematical Programming*, 120:347–380, 2009.

[3] R. Baldacci, N. Christofides, and A. Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115:351–385, 2008.

[4] R. Baldacci, E. Bartolini, A. Mingozzi, and R. Roberti. An exact solution framework for a broad class of vehicle routing problems. *Computational Management Science*, 7: 229–268, 2010.

[5] R. Baldacci, E. Bartolini, A. Mingozzi, and A. Valletta. An exact algorithm for the period routing problem. *Operations Research*, 59:228–241, 2011.

[6] R. Baldacci, A. Mingozzi, and R. Roberti. New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research*, 59:1269–1283, 2011.

[7] R. Baldacci, A. Mingozzi, and R. Wolfler-Calvo. An exact method for the capacitated location-routing problem. *Operations Research*, 59:1284–1296, 2011.

[8] J. M. Belenguer, E. Benavent, C. Prins, C. Prodhon, and R. Wolfler-Calvo. A branch-and-cut algorithm for the capacitated location routing problem. *Computers & Operations Research*, 38:931–941, 2011.

[9] N. Christofides and S. Eilon. An algorithm for the vehicle dispatching problem. *Operational Research Quarterly*, 20:309–318, 1969.

[10] N. Christofides, A. Mingozzi, and P. Toth. *The Vehicle Routing Problem*. Wiley, Chichester, UK, 1979.

[11] N. Christofides, A. Mingozzi, and P. Toth. Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations. *Mathematical Programming*, 20:255–282, 1981.

[12] C. Contardo, J.-F. Cordeau, and B. Gendron. A computational comparison of flow formulations for the capacitated location-routing problem. Technical Report CIRRELT-2011-47, Université de Montréal, Canada, 2011.

[13] C. Contardo, J.-F. Cordeau, and B. Gendron. A branch-and-cut-and-price algorithm for the capacitated location-routing problem. Technical Report CIRRELT-2011-44, Université de Montréal, 2011.

[14] J.-F. Cordeau and M. Maischberger. A parallel iterated tabu search heuristic for vehicle routing problems. *Computers & Operations Research*, 39:2033–2050, 2012.

[15] J.-F. Cordeau, M. Gendreau, and G. Laporte. A tabu search heuristic for periodic and multi-depot vehicle routing problem. *Networks*, 30:105–119, 1997.

[16] G. B. Dantzig, D. R. Fulkerson, and S. M. Johnson. Solution of a large-scale traveling salesman problem. *Operations Research*, 2:393–410, 1954.

[17] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40:342–354, 1992.

[18] M. L. Fisher. Optimal solution of vehicle routing problems using minimum k-trees. *Operations Research*, 42:626–642, 1994.

[19] R. Fukasawa, H. Longo, J. Lysgaard, M. Poggi de Aragão, M. Reis, E. Uchoa, and R. F. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming Series A*, 106:491–511, 2006.

[20] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[21] R. E. Gomory. Outline of an algorithm for integer solutions to linear problems. *Bulletin of the American Mathematical Society*, 64:265–279, 1958.

[22] D. Houck, J. Picard, M. Queyranne, and R. Vemuganti. The travelling salesman problem as a constrained shortest path problem: theory and computational experience. *Opsearch*, 17:93–109, 1980.

[23] S. Irnich and D. Villeneuve. The shortest path problem with resource constraints and k-cycle elimination for k$\geq$3. *INFORMS Journal on Computing*, 18:391–406, 2006.

[24] M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, 56:497–511, 2008.

[25] N. Kohl. *Exact methods for time constrained routing and related scheduling problems*. PhD thesis, Institute of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 1995.

[26] J. Larsen. *Parallelization of the Vehicle Routing Problem with Time Windows*. PhD thesis, Department of Mathematical Modelling, Technical University of Denmark, 1999.

[27] J. Lysgaard. *CVRPSEP: A package of separation routines for the capacitated vehicle routing problem*, December 2003.

[28] J. Lysgaard, A. N. Letchford, and R. W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100:423–445, 2004.

[29] J. Renaud, G. Laporte, and F. F. Boctor. A tabu search heuristic for the multi-depot vehicle routing problem. *Computers and Operations Research*, 23:229–235, 1996.

[30] G. Righini and M. Salani. New dynamic programming algorithms for the resource constrained elementary shortest path problem. *Networks*, 51:155–170, 2008.

[31] T. Vidal, T. G. Crainic, M. Gendreau, N. Lahrichi, and W. Rei. A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems. *Operations Research*, Forthcoming, 2012.